Mini projet JAVA

Etudiants I1 groupe 49

Sébastien FRANCOIS
Cyril HAMELIN

Responsable de projet

Olivier Camp

Sommaire

1) CAHIER DES CHARGES UTILISATEUR 3

2) ETAT D'AVANCEMENT 4

3) POURQUOI REALISER UN LOGICIEL D'EDITION ? 5

3.1) SPECIFICATIONS PROGRAMMEUR 6

4) TECHNIQUES UTILISEES 7

4.1) LA SERIALISATION 7
4.2) LE MODELE OBJET POUR DES TYPES DE GROUPES / D'UNITES 8
4.3) DIAGRAMMES UML 9
4.9) UTILISATION D'ECLIPSE 11
4.10) TRAVAIL COLLABORATIF PAR CVS 12

5) MODE D'EMPLOI DU LOGICIEL : LE CREATEUR D'ARMEE 13

5.1) RAPPEL SUR LE BUT DE CE LOGICIEL: 13
EXEMPLE DE CREATION D'UN GROUPE 14
5.2) MENU : CHARGER 17
5.3) MENU : SAUVEGARDER 17

6) REPARTITION DU TRAVAIL 18

7) CONCLUSION 18

8) ANNEXE: WARHAMMER LE JEU 19

8.1) QU'EST CE QUE WARHAMMER LE JEU ? 19
8.2) QU'EST CE QU'UN LIVRE D'ARMEE ? 19
8.3) DE QUOI A-T-ON BESOIN POUR JOUER ? 19
8.4) QU'EST CE QU'UNE LISTE D'ARMEE ? 20
8.5) QU'EST CE QU'UNE ARMEE? 21

1) Cahier des charges utilisateur

- Réaliser un logiciel permettant de faciliter la création d'une liste d'armée pour le jeu Warhammer.
- Spécifications : respecter le format d'une armée du jeu, pouvoir afficher en temps réel la liste d'armée et la photo de l'armée créée à partir de chacune des photos d'unités.
- Toutes les possibilités d'une armée doivent être respectées avec cet éditeur.
- Pouvoir enregistrer, reprendre et imprimer une liste créée.

2) Etat d'avancement

Etude préalable	Avancement
Elaboration du cahier des charges	**COMPLET**
Modélisation	**COMPLET**
Etude de l'implémentation	**COMPLET**

Programmation d'une armée	Avancement
Modèle	**COMPLET**
Vue	**COMPLET**

Programmation du créateur d'armée	Avancement
Gestion armée / groupe	**COMPLET**
Gestion unité / option	**PARTIEL**
Sérialisation	**COMPLET**
Association d'image aux groupes et unités	**COMPLET**

Les options existent dans le modèle et fonctionnent, cependant l'onglet pour les éditer n'existe pas encore.

Nous avons commencé l'application créateur de liste, mais nous ne sommes pas rendus assez loin pour pouvoir montrer un quelconque résultat.

Les fonctions à assurer sont :

- lecture d'une armée sérialisée
- création d'une liste par sélection d'unité dans l'armée de référence
- calcul des coûts, respect des règles
- affichage d'une superposition des images des unités pour voir l'armée

3) Pourquoi réaliser un logiciel d'édition ?

Réaliser une liste d'armée « à la main » est long et contraignant. Il faut constamment chercher dans le livre d'armée les descriptions de chaque élément.

Il existe une communauté de joueurs de Warhammer en France et plus particulièrement sur Angers (une cinquantaine de personnes), et des tournois sont régulièrement organisés. Ils se réalisent exactement comme une rencontre sportive, à la manière d'une ronde suisse. Cette communauté de joueurs Angevins possède un forum Internet http://www.letrolldelouest.com/.

Avec un outil tel que cet éditeur de liste d'armée, il serait simple de mettre l'application JAVA en ligne sur ce site. Ainsi lors de la préparation d'un tournoi, les joueurs pourraient directement préparer leur liste d'armée suivant un même format, et envoyer par mail le résultat à l'arbitre du tournoi qui est chargé de l'organisation et de la vérification des listes d'armées. C'est donc concrètement que ce logiciel peut aider à standardiser une liste d'armée, avec un gain de temps important pour les joueurs et pour lui-même.

Chaque joueur possède ses propres figurines qu'il a assemblées et peintes. L'intérêt de ce logiciel est de pouvoir insérer ses propres photos pour la visualisation de l'armée.

Conclusion:

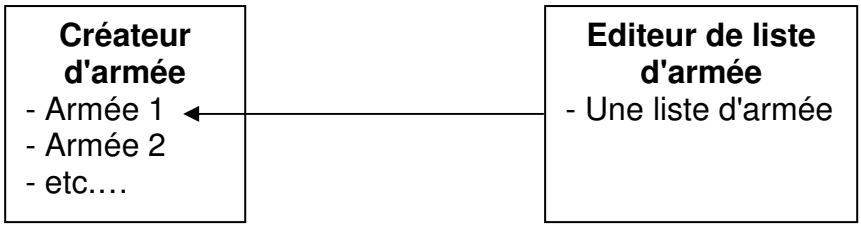
- ✓ Liste d'armée plus rapide à faire qu'à la main.
- ✓ Possibilité de mettre en ligne le logiciel sur une page HTML
- ✓ Possibilité de reprendre très facilement une liste déjà créée
- ✓ Visualisation directe de l'armée
- ✓ Personnalisation : chaque utilisateur peut créer sa propre base de données simplement avec ses propres photos de figurines

3.1) Spécifications programmeur

En tant que programmeur, il est nécessaire de traduire le cahier des charges utilisateur en besoin technique. L'utilisateur souhaite avoir en base de données tous les livres d'armée du jeu. Et à partir de l'une d'entre elle, il veut pouvoir faire une liste d'armée.

Il faut donc réaliser le logiciel en 2 parties. Une première consiste à créer un logiciel d'accès à cette base de données, en pouvant enregistrer, créer et reprendre n'importe quel livre d'armée. Dans un second temps un deuxième logiciel s'occupera de l'assemblage d'une liste d'armée, en piochant dans les livres d'armée disponibles grâce au premier logiciel.

- Nécessité de faire 2 logiciels



Logiciel 1 Logiciel 2

Une armée comprend tous les groupes disponibles, et le joueur vient sélectionner parmi elle ceux qu'il désire pour les mettre dans sa liste d'armée. On constate donc qu'il n'y a pas de différence conceptuelle entre les deux. Nous avons donc choisi le même modèle pour définir les deux. Une armée est en fait une liste d'armée qui comprend tous les groupes de l'armée.

Pour programmer ces logiciels, nous utilisons une représentation MVC. Ainsi le modèle est le même pour les 2 logiciels, seuls la vue et le contrôleur changent.

4) Techniques utilisées

4.1) La sérialisation

Les différents groupes, unités et leurs options disposent de propriétés spécifiées précisément dans les guides de jeu Warhammer.

Le cas d'utilisation typique se déroule donc de la manière suivante :

- dans un premier temps, le joueur utilise le créateur d'armée pour saisir chaque classe d'unité de son armée.
  Il remplit l'ensemble des champs concernant l'unité (nom, coût, type).
  Les données sont enregistrées, ces unités seront ensuite assimilées à des unités de base qui serviront à créer une armée de jeu.

- Dans la phase de préparation d'une partie de Warhammer, le joueur utilise le créateur de liste d'armée et vient sélectionner, parmi cette librairie d'unités, celles qu'il souhaite utiliser, tout ceci dans le respect des règles et du coût total qui lui est imposé.

Face à ce projet, nous avions premièrement pensé à saisir les données dans une base MySQL, ceci aurait permis de pouvoir facilement partager ses unités avec d'autres joueurs. Mais dans la pratique, cela rendait son utilisation compliquée lors d'une compétition (besoin d'avoir un serveur MySQL sur la machine…). Nous avons donc abandonné cette solution.

Les langages orientés objet proposent une solution efficace pour la sauvegarde / restauration d'objets à propriétés multiples, il s'agit de la sérialisation. Sérialiser consiste à transformer un objet instancié en un fichier dont les données seront donc conservées au-delà de l'exécution du programme.
Par la suite, on pourra ouvrir ce fichier, et instancier de nouveau les objets. Il est aussi possible d'utiliser la sérialisation pour transférer par exemple des objets à travers un réseau, mais cela dépasse le cadre de notre projet, et nous n'aborderons ici que la sérialisation vers un flux stocké dans un fichier.

En java, la sérialisation se fait grâce aux classes de java.io, ObjectOutputStream et ObjectInputStream. Elles sérialisent dans un format « binaire ». Nous avons donc voulu voir s'il était possible d'utiliser un format plus standard tel que le XML. Il existe bien des classes XMLEncoder et XMLDecoder, mais celles-ci sont prévues pour fonctionner avec des java beans.
Il est possible de surcharger les fonctions de sérialisation standards pour utiliser un format de sauvegarde plus personnalisé, mais cela aurait très vite compliqué le code.

L'ensemble des propriétés d'une armée, des groupes et unités qu'elle contient est sauvegardé dans un fichier .swa. Les images quant à elles, ne pouvant pas être sérialisées aisément, nous avons opté pour une solution simple :

1. on créé un répertoire du nom du fichier sauvegardé,
2. on parcourt l'ensemble des groupes et des unités

3. si une image est associée à l'un d'entre eux, alors on la sauvegarde dans le répertoire, elle est nommée selon ses indices

Par exemple :

- logo du premier groupe : **répertoire/group0.png**
- image de la deuxième unité du 3ème groupe : **répertoire/group2unit1.png**

Lors de la desérialisation du fichier swa, il est donc facile de réaffecter chaque image à son objet.

Le code correspondant à ces explications se situe dans controller.armyBuilder : MenuAdapter.java

4.2) Le modèle objet pour des types de groupes / d'unités

Les règles du jeu Warhammer prévoient qu'une armée peut être constituée de 5 différents groupes. Ces groupes sont rigoureusement identiques. Lors de la modélisation, nous nous sommes donc posée la question de savoir s'il fallait utiliser une propriété de classe qui stockerait le type de groupe, ou bien si nous pouvions créer une classe pour chaque type qui hériterait d'un groupe abstrait.

Nous avons opté pour la seconde solution, qui nous semblait plus cohérente avec l'orientation objet et notre modèle.

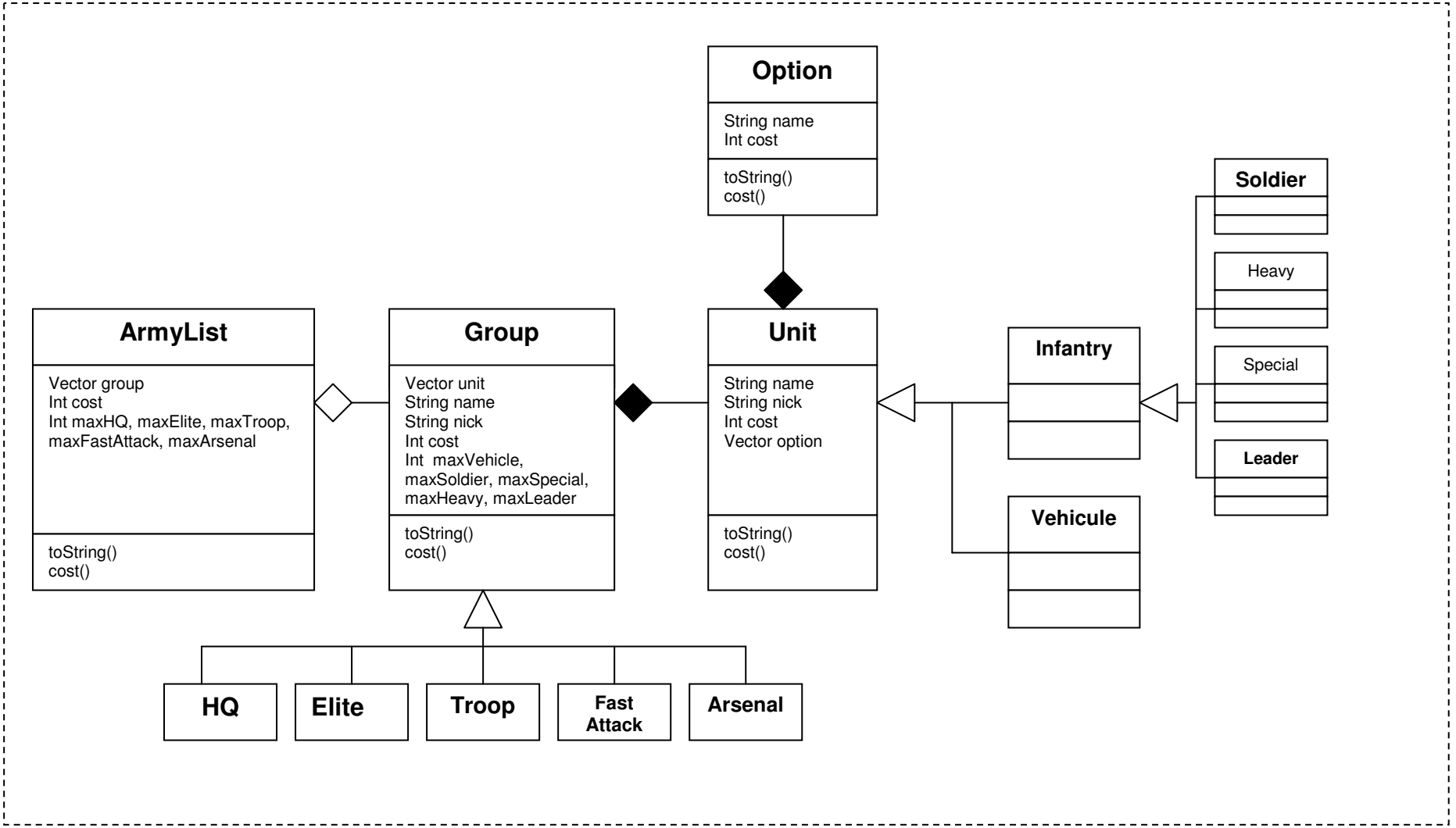
Il s'est alors posé un problème :

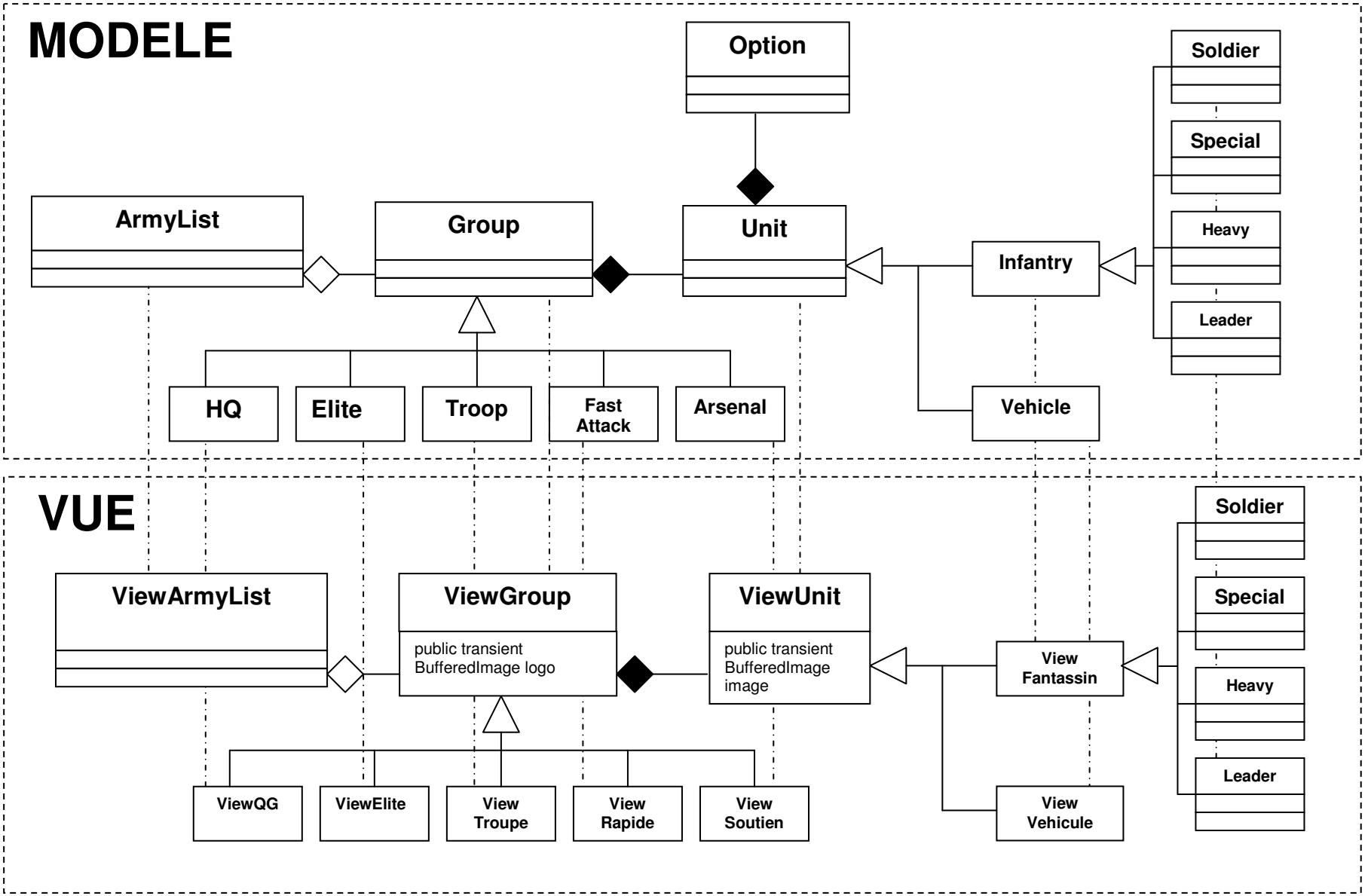
> ***comment, par exemple, changer un groupe Quartier Général en Troupe après que celui-ci ait été instancié ?***

Nous avons tout d'abord cherché à cloner la classe mais le type restait le même. Par la suite, nous avons mis en place un constructeur spécial, dédié à la duplication. Implémenté dans le groupe abstrait, Il extrait l'ensemble des champs d'un groupe et les transmet à un constructeur. Ce constructeur spécial est propagé dans chaque classe fille, nous pouvons donc dès lors dupliquer un groupe en changeant son type.

De manière plus claire, dans cet exemple, nous transmettons au constructeur de Troupe l'objet instancié de type Quartier Général, pour sa duplication.
Celui-ci le passe au constructeur de sa classe mère qui en extrait les champs et créé un Groupe. Ce groupe est alors transformé en Troupe, et ses propriétés sont identiques à celles de l'objet d'origine.

4.3) Diagrammes UML



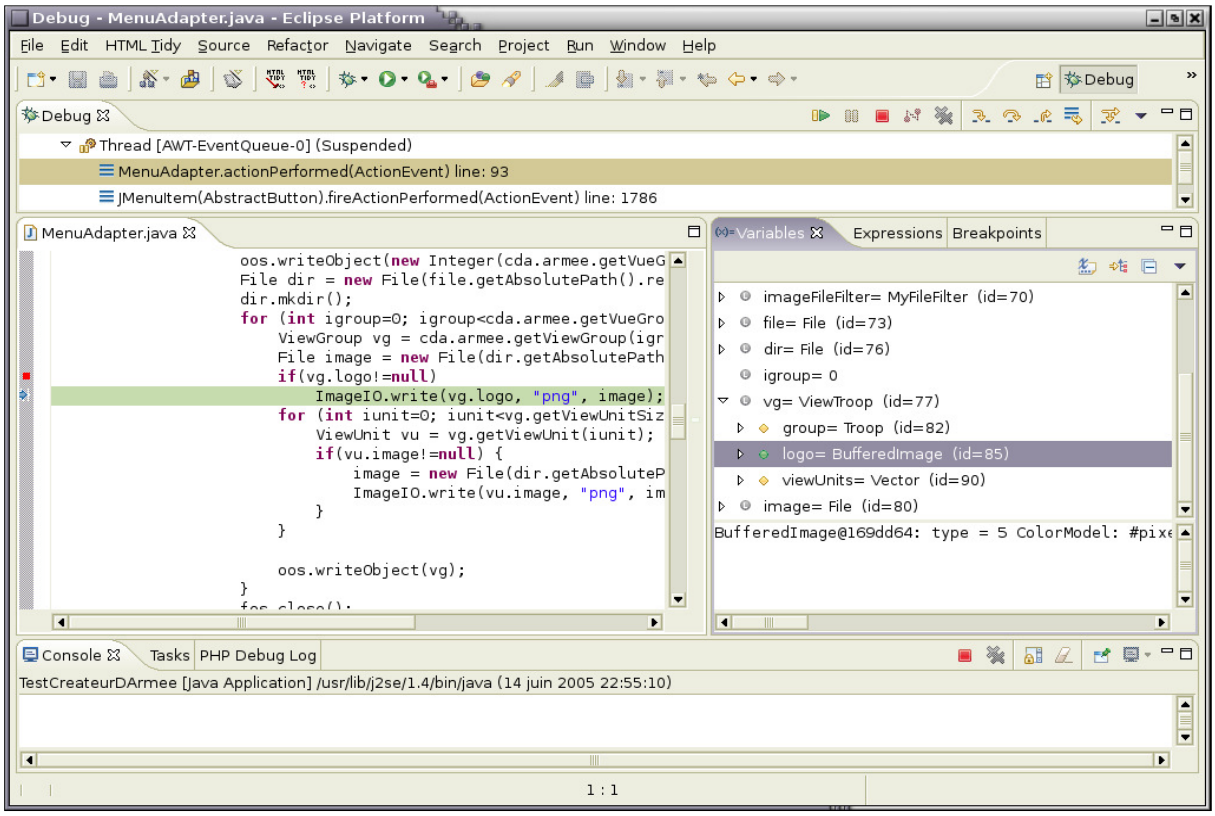

MODELE
Option
Soldier
Special
Heavy
Leader
ArmyList
Group
Unit
Infantry
HQ
Elite
Troop
Fast Attack
Arsenal
Vehicle
VUE
ViewArmyList
ViewGroup
public transient BufferedImage logo
ViewUnit
public transient BufferedImage image
View Fantassin
ViewQG
ViewElite
View Troupe
View Rapide
View Soutien
View Vehicule

4.9) Utilisation d'Eclipse

Nous avons choisi de développer ce projet sur Eclipse, car nous l'avions utilisé durant la mise au point du Bureau d'études et souhaitions en poursuivre la découverte.

L'auto complétion, ainsi que la validation du code en cours de frappe, nous ont permis de gagner du temps, puis par la suite, l'exécution en mode déboguage nous a permis de suivre finement l'exécution du programme.

Exemple d'exécution en pas à pas :

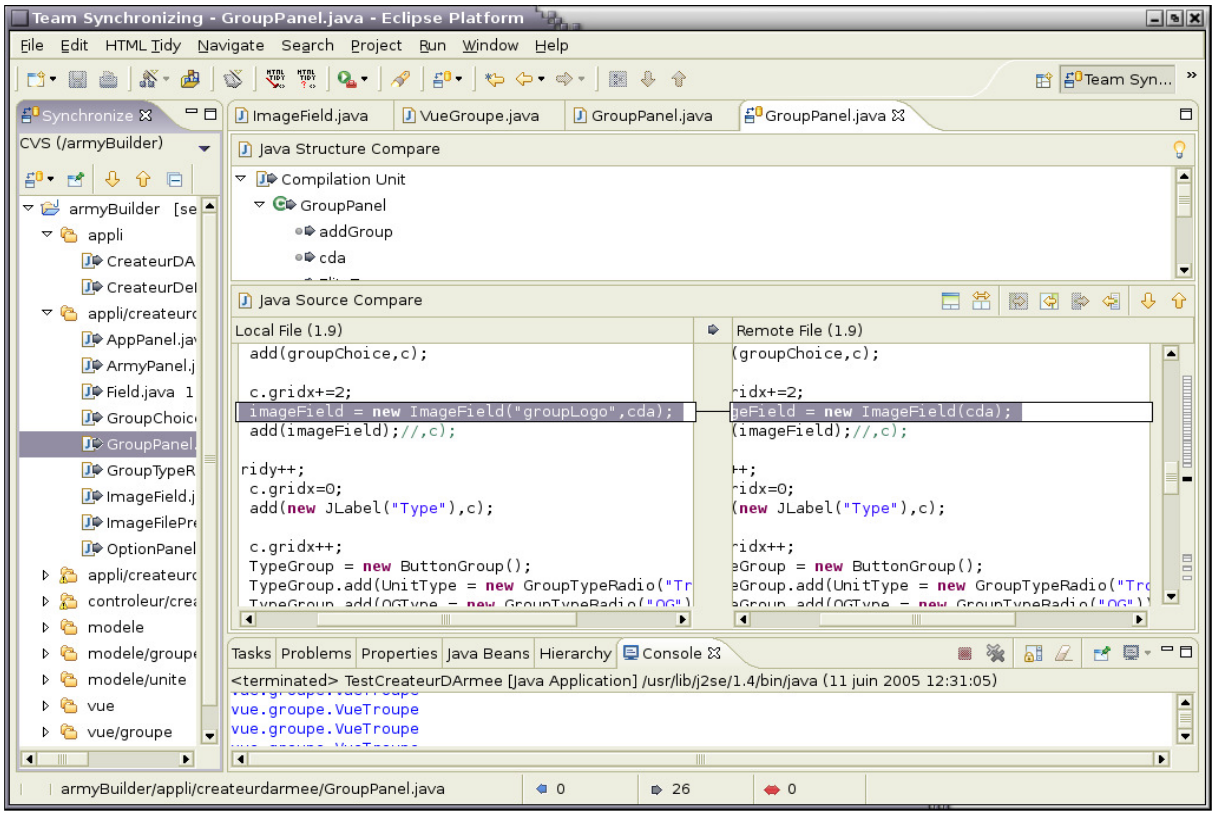


Après une demande d'enregistrement de l'armée, l'ordinateur vient de sauvegarder les données relatives à l'armée et poursuit avec les groupes.
Le programme enregistre les données relatives au premier groupe de l'armée (igroup=0).
Il vient de tester si ce groupe avait un logo. Nous pouvons voir en mémoire dans le panneau de droite qu'une BufferedImage est bien affectée à la propriété logo du ViewTroop.

4.10) Travail collaboratif par CVS

Dès lors que plusieurs personnes travaillent sur des sources communes, il peut très facilement survenir des problèmes de version entre les fichiers. Chacun travaille donc sur une copie propre des sources et la mise en commun fait souvent perdre du temps.

Nous avons donc, dès le départ, mis en place un CVS sur une machine personnelle, ainsi que deux comptes, et avons configuré Eclipse pour qu'il puisse y accéder.
La gestion du CVS d'Eclipse nous a été très utile pour travailler en parallèle, sans risquer d'écraser le travail de l'autre. Nous avons aussi pu garder une trace, fichier par fichier de l'évolution du projet.



Dans la perspective Team Synchronisation, nous pouvons voir dans le panneau de gauche l'ensemble des fichiers modifiés, les flèches grises indiquent que ce sont des modifications faites localement. Il n'y a pas de fichier plus récent dans le CVS (compteur à 0 à côté de la flèche bleue en bas), il n'y a pas non plus de collision (double flèche rouge).

Le panneau de droite nous permet d'observer la différence entre le fichier local et celui contenu dans le CVS. Ici, le constructeur reçoit un paramètre supplémentaire.

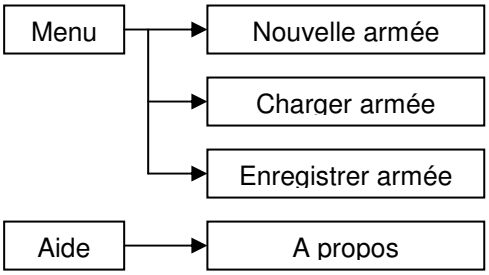
5) Mode d'emploi du logiciel : Le Créateur d'armée

Dans l'état d'avancement actuel du projet, le premier logiciel est terminé. Voici donc son mode d'emploi.

5.1) Rappel sur le but de ce logiciel:

Il s'agit de rentrer les données d'une armée (image, profils de figurine, appartenance à un groupe, etc.…) dans un fichier de sauvegarde. Ces fichiers d'armées sont ensuite utilisés comme base de données par le deuxième logiciel Editeur de liste d'armée.

Voici le plan des menus du logiciel:



- ✓ Créer une armée:

Il s'agit dans un premier temps de créer une armée. Nous verrons par la suite comment sauvegarder puis charger une armée. Après avoir sélectionné Menu>Nouvelle Armée, des onglets sont disponibles. Ils correspondent au schéma d'une armée standard. Dans l'exemple qui suit, on va créer la base de données pour l'armée de Space Marine donnée en exemple dans ce rapport.

- ✓ Créer un groupe

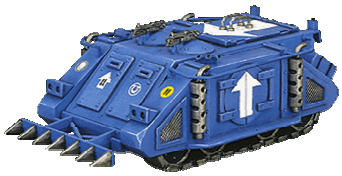
Tout ce qui concerne le groupe est dans l'onglet groupe. C'est ici qu'on crée et qu'on ajoute chaque groupe de l'armée.

- ✓ Créer une unité

Tout ce qui concerne l'unité est dans l'onglet unité.

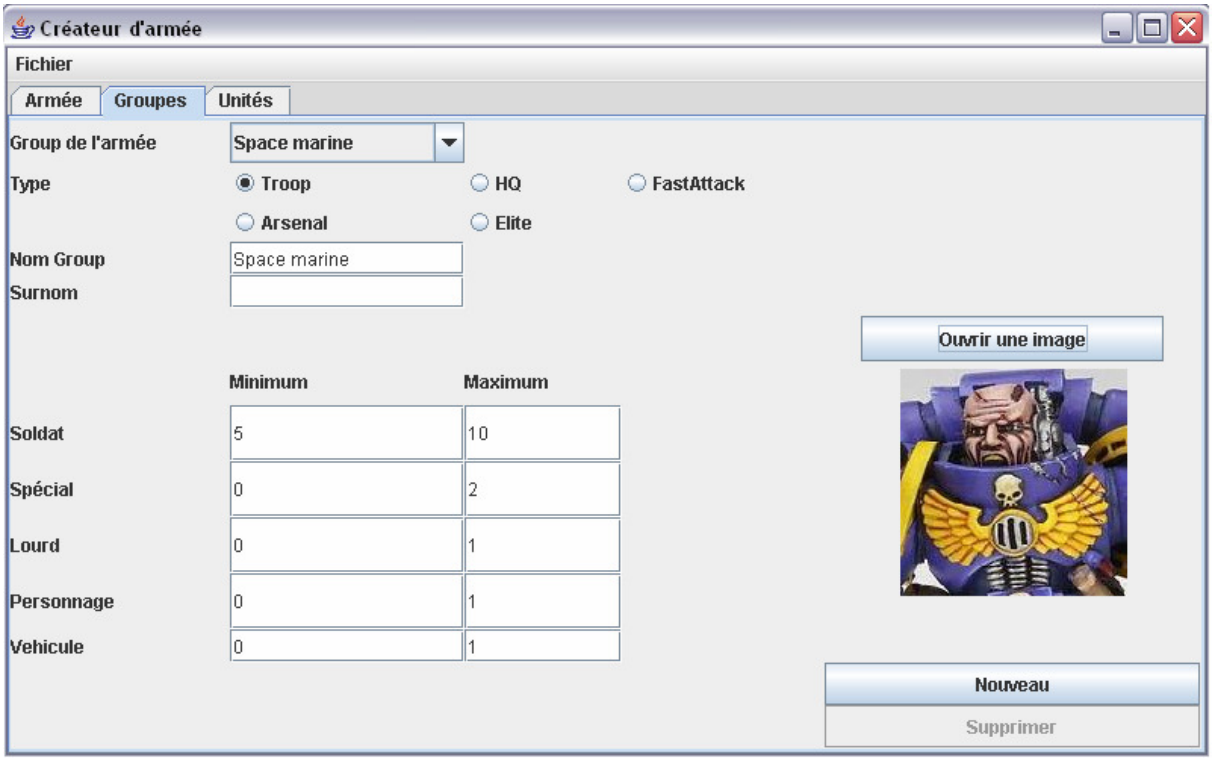
Exemple de création d'un groupe

En terme de jeu et pour un groupe, un joueur a le choix parmi plusieurs unités pour constituer son groupe. Il faut donc les identifier selon le schéma du logiciel pour pouvoir les entrer dans base de données.

Groupe	Unité Type:nom	Image associée	Coût
Space Marine	Soldat: Space marine		15pts
	Spécial: lance flamme		20pts
	Lourd: Canon laser		30pts
	Lourd: Bolter lourd		25pts
	Lourd: Lance plasma		30pts
	Lourd: Lance missile		25pts
	Sergent		30pts
	Véhicule: Transport		50pts
	Véhicule: Transport lourd		75pts

Type du groupe: Troop
Maximum de soldat dans un groupe: 10
Maximum de spéciaux dans un groupe: 1
Maximum de lourd dans un groupe: 1
Maximum de sergent dans un groupe: 1
Maximum de véhicule dans un groupe: 1

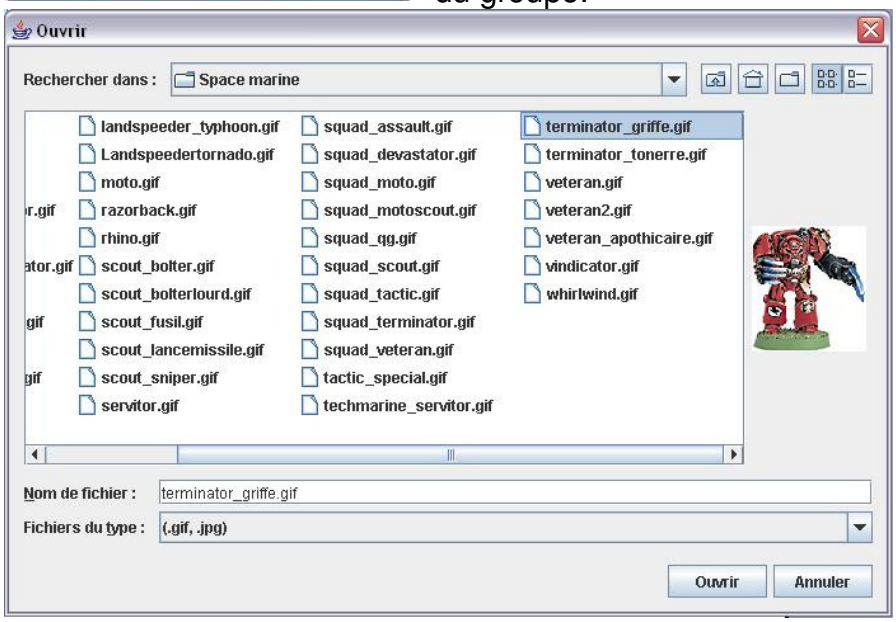
Voici l'exemple pour créer ce groupe:



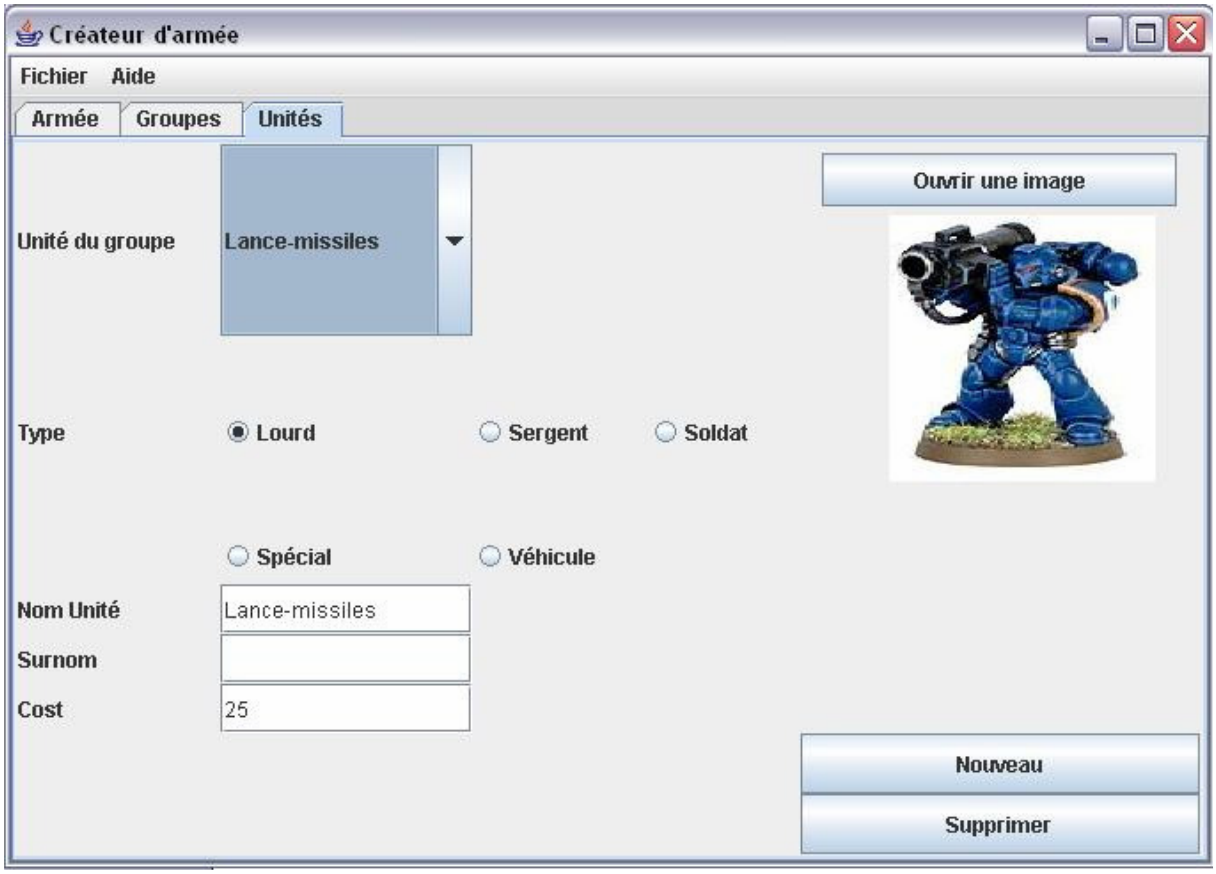
Ajoute un nouveau groupe à l'armée, il possède toutes les unités du dernier groupe créé (cela évite à l'utilisateur de tout ressaisir si le nouveau groupe est proche du dernier créé)

Supprime le dernier groupe ajouté.

Ouvre la boite de dialogue pour aller chercher l'image du groupe:



Ensuite il faut créer chaque unité de ce groupe, en voici un exemple:



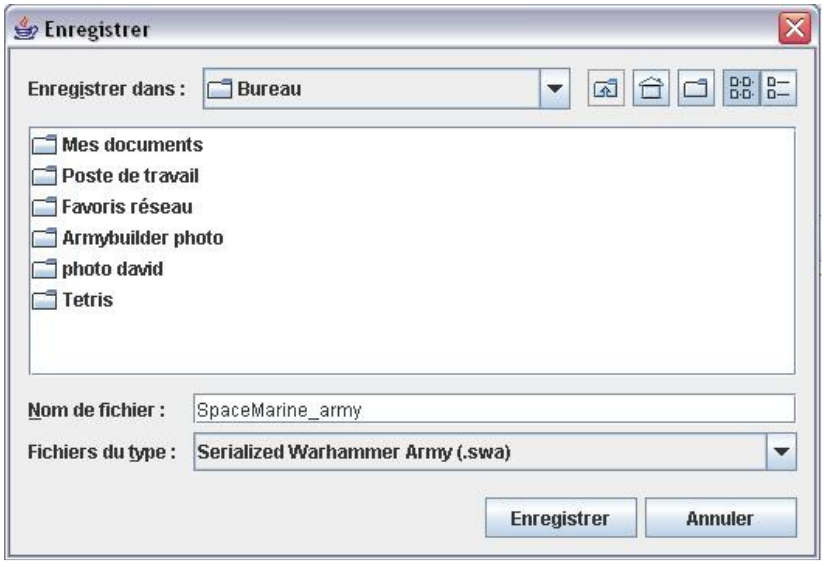
Ajoute une nouvelle unité avec les mêmes caractéristiques que la dernière (cela évite à l'utilisateur de tout ressaisir si l'unité nouvelle est proche de la dernière créée)

Supprime la dernière unité ajoutée.

Ouvre la boite de dialogue pour aller chercher l'image.

5.2) Menu : Charger

5.3) Menu : Sauvegarder



6) Répartition du travail

Tâches	Etudiant
Cahier des charges	**Cyril**
Modélisation UML	**Cyril**
Etude de l'implémentation	**Sébastien**
Création des modèles et de leurs vues	**Cyril**
Création de l'application et des contrôleurs	**Sébastien** / **Cyril**
Mise en place du CVS	**Sébastien**
Sérialisation	**Sébastien**
Rapport	**Sébastien** / **Cyril**

7) Conclusion

Nous vous sommes reconnaissants de nous avoir autorisés à investir ces séances de mini-projet dans un projet qui nous intéressait. Pendant sa mise au point, nous avons pu découvrir de nombreuses astuces sur le fonctionnement d'Eclipse, mais aussi sur le langage, mettant en place une sérialisation, la possibilité de « transtyper » des classes.

Le langage Java nous apparaît comme un langage structuré, et dont l'ensemble des classes fournies dans javax et java permet de créer très rapidement des applications simples. En revanche, nous avons pu constater une certaine lenteur dès lors que l'application tournait en mode débogage dans Eclipse.

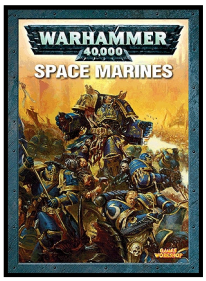
Il serait intéressant que l'ESEO fournisse un CVS accessible de l'extérieur.

8) Annexe: Warhammer le jeu

8.1) Qu'est ce que Warhammer le jeu ?

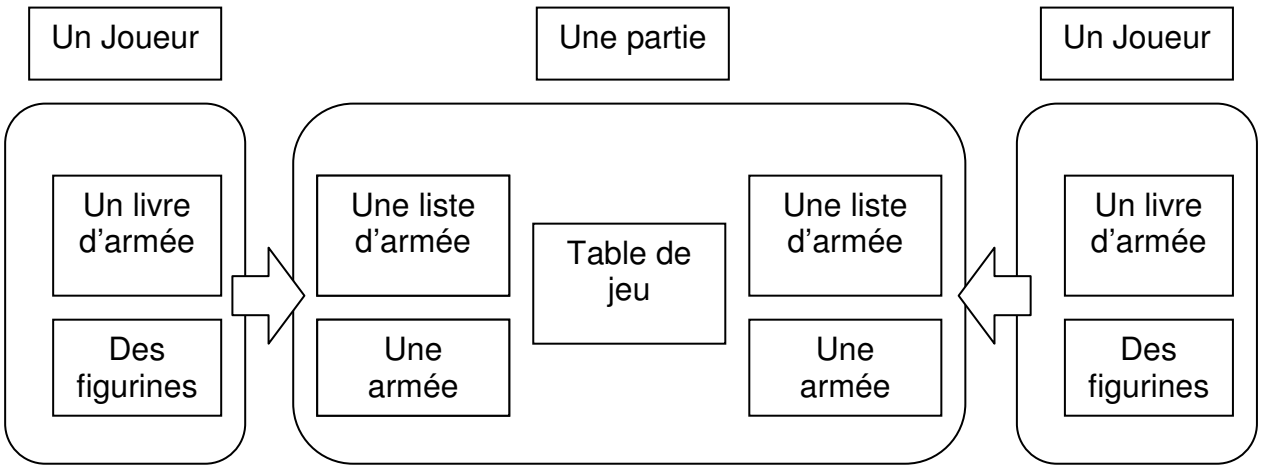
C'est un jeu qui oppose 2 joueurs, chacun possédant un ensemble de figurines appelé une armée. Les joueurs jouent chacun leur tour, avec des règles de jeu bien définies. Un joueur constitue une armée à partir d'un livre d'armée. Chaque figurine de l'armée possède un profil propre et un coût en points. Une partie est jouée avec un coût total de points par armée fixé par les 2 joueurs.

8.2) Qu'est ce qu'un livre d'armée ?



Un livre d'armée contient tous les profils et les coûts des figurines d'une armée. Il est spécifique à une armée. Le joueur sélectionne des figurines parmi celle du livre d'armée jusqu'à arriver à son total de points fixé pour la partie. Il assemble ce qu'on appelle une liste d'armée.

8.3) De quoi a-t-on besoin pour jouer ?



8.4) Qu'est ce qu'une liste d'armée ?

Une liste d'armée est un document papier qui contient la liste des figurines sélectionné par un joueur. Elle récapitule le coût et les options de chaque figurine de l'armée. Le joueur s'y réfère tout au long d'une partie. Elle est la représentation « papier » de l'ensemble des figurines utilisées pour la partie. On peut voir sur la photo ci-dessus chaque joueur présentant leur liste d'armée.

Liste d'armée	Armée

Armée de Space Marine

Quartier Général:

Seigneur (60), armure terminator (25), paire de griffe éclair (30). 115pts

Elite :

5 terminators (200), 1 lance-flammes (15). 215pts

Troupe :

5 Spaces marines (70), Lourd : Lance missile (15), Spécial : lance flamme (10). 95pts

5 Scouts (60), 1 sniper (10). 70pts

Attaque Rapide :

3 motos Space marines (135), 1 lance grenade (10). 145pts

Soutien :

Artillerie (125): amélioration de coque (30). 155pts

Total 1000pts.

8.5) Qu'est ce qu'une armée?

Une armée est constituée de groupes, qui eux-mêmes sont constitués d'unités.